<u>**AMENDMENTS TO THE SPECIFICATION:**</u>

Please amend paragraphs 0037-0038 as follows:

[0037] In reference now to FIG. 4, the interactions between a GPPI using factory classes are illustrated according to embodiments of the present invention. A generic TSY 402 and cellular modem software (CSW) 408 communicate via a GPPI 410 that includes an abstract primitive factory 404 and an abstract passing handler 406. When the TSY 402 initiates a service request directed for the cellular modem software 408, the TSY 402 first requests 412 that the abstract primitive factory 404 creates the appropriate primitive. The abstract primitive factory 404 then creates 414 the appropriate abstract primitive 416 for the task. The TSY 402 then uses the abstract primitive ~~414~~ <u>416</u> to pass 418 the service request to the cellular modem software 408. At this point, the abstract primitive ~~414~~ <u>416</u> externalizes 420 a concrete primitive for use by the cellular modem software 408.

[0038] Generally, externalization refers to the process of recording the object state in a stream of data. In other words a chunk of data is created in the format defined by the interface of cellular modem software. So, in this example, the abstract primitive ~~414~~ <u>416</u> may assume the characteristics of one or more concrete primitives particular to the cellular modem software 408. The externalization 420 may occur at a concrete primitive factory (not shown) that inherits from the abstract primitive factory 404. Similarly, internalization refers to the process of instantiating an object from a chunk of data received from the cellular modem software 408. Internalization is performed as well as externalization by a concrete primitive factory (e.g., concrete primitive factory 314 in FIG. 3) that is hidden by the abstract <u>primitive factory</u> interface 404. The concrete primitive factory creates a new instance of a concrete primitive that is used by GPPI and/or TSY through their abstract primitive interface(s).